Utilizing an Enhanced Cellular Automata Model for Data Mining

Omar Adwan[1], Ammar Huneiti[1], Aiman Ayyal Awwad[2], Ibrahim Al Damari[1], Alfonso Ortega[3], Abdel Latif Abu Dalhoum[1],Manuel Alfonseca[3]

Abstract - Data mining deals with clustering and classifying large amounts of data, in order to discover new knowledge from the existent data by identifying correlations and relationships between various data-sets. Cellular automata have been used before for classification purposes. This paper presents a cellular automata enhanced classification algorithm for data mining. Experimental results show that the proposed enhancement gives better performance in terms of accuracy and execution time than previous work using cellular automata.

Keywords: *Cellular Automata, Clustering, Classification, Data Mining, Moore Neighborhood.*

I. Introduction

A cellular automaton (CA) is a discrete mathematical model with three main components; namely, a finite automaton, a regular lattice (grid) not necessarily finite, and a neighborhood rule that defines the set of neighboring cells for every position in the grid. The global behavior of a CA may be described locally because each finite automaton in the grid takes the states of its neighbors as input. However, cellular automata with simple local behavior may give rise to complex dynamic systems. The set of particular states of all the automata in the grid of a CA at a given time is called a configuration. The grids can be seen as matrices of states with a given dimension. The most common grids are one and two dimensional grids.

Cellular automata have been successfully used in practice in many different ways as simulation tools for a wide variety of disciplines, including physical modeling and simulation [1], biology [2], fluid dynamics [3], pattern recognition [4], logical organization behind self-reproduction [5], traffic simulation [6] [7], edge detection [8], and urban development simulation [9] [10]. In the theoretical domain, they have been used as parallel computer abstract architectures [11] [12] [13] [14]. In [15] and [16] cellular automata were connected with formal languages as a standard method to study other decentralized spatially extended systems. CA's have also been used as an alternative method to solve differential equations and to simulate several physical systems where differential equations are useless or difficult to apply [14].

Data mining refers to the process of analyzing huge data bases in order to find useful patterns. As in knowledge discovery, or statistical analysis, in data mining one of the most important applications is prediction. Data prediction has two main approaches, supervised and unsupervised [17]. Classification is the process of assigning an item the class to which it belongs, so it is a prediction process based on a rule. To classify the data, one must find rules that group the provided data instances into appropriate classes [18] [19] [20].

Cellular automata can be used successfully for data mining [21] [22] [23], because all decisions made locally depend on the state of each cell and the states of the neighboring cells.

In this paper we have used as basic data the same set of fRMI data that we classified in a previous paper [23] using an uni-dimensional CA. Here we are classifying those data with a two-dimensional CA which enhances in several ways the model proposed in [21]. This paper is an extended version of a short communication presented to an international conference [24].

II. Cellular Automata

There are many different types of cellular automata, depending on the differences of their components. These components are the states of the cell, the geometrical form of the lattice, the neighborhood of a cell, and the local transition function [1] [25].

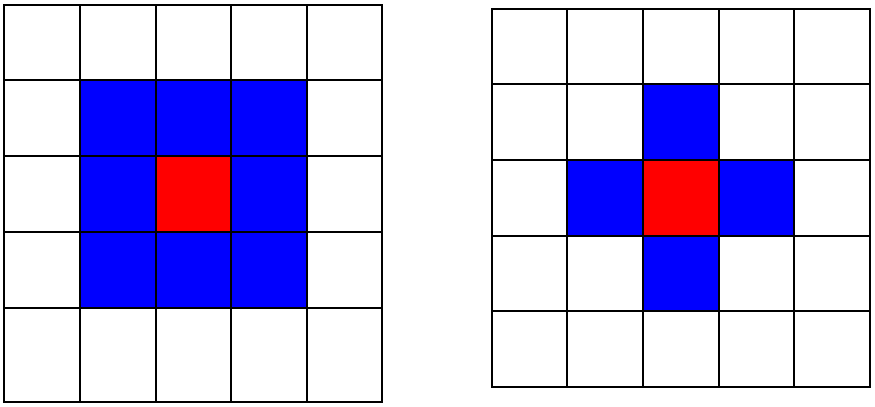
One of the best known cellular automata is the game of life, introduced by John Conway [26]. The game of life is a very simple cellular automaton that has been proved to be computationally complete, being able (in principle) to perform any computation which can be done by digital computers, Turing machines or neural networks.

The cellular automaton associated to the game of life is defined thus:

- The grid is rectangular and potentially infinite.

- The set of neighbors to a point in the grid consists of the point itself plus the eight adjacent points in the eight main directions in the compass (Moore's neighborhood).
- Each finite automaton has two states: empty (also called dead, represented by a zero or a *space* character) and full (also called alive, represented by a one or a star symbol *). The set of states is thus represented by the two Boolean numbers {0,1} or the two characters *space* and '*'.
- The transition function is defined by the following simple rules:
  - If the automaton associated to a cell is in the empty state, it goes into the full state if and only if the number of its neighbors in the full state is exactly three.
  - If the automaton associated to a cell is in the full state, it goes into the empty state if and only if the number of its neighbors in the full state is less than two or more than three.
  - In any other case, the automaton remains in the same state.
- Each time step is called a "generation". The set of all the cells alive at a given time step is called the "population".

The fact that the grid is potentially infinite makes the game of life difficult to implement. However, restricted versions, associated to a grid of finite dimensions, are very simple, at the cost of losing computational completeness.
The transition function defines the next state of the cell depending on its current state and the states of its neighbors (which act as the input to the finite automaton in the cell). There are different ways to define the neighborhood. The most common neighborhoods are Moore Neighborhood and von Neumann Neighborhood as shown in Fig.1 below.



(a) Moore Neighborhood **(b)Von Neumann Neighborhood**

Fig. 1 Common Neighborhoods

Every cell uses the same update rules, which are applied to all the cells in the lattice simultaneously and synchronously. The update rule depends only on just the neighbors of each cell, so the process is local.

III. Data Mining

Classification is a supervised technique. This means that prior knowledge about the data is used to classify unknown data. Given an item, we want to determine to which class it belongs. To perform a classification, one must find a function or a set of rules that classify the given test data samples into specific groups. This function is the output of a training technique that uses the training data samples [17, 27]. In a formal way:

- Let the n dimensional space of real numbers R^n be our data universe, with points $x \in R^n$.
- Let S be a sample set such that $S \subset R^n$.
- Let $f : R^n \to \{-1,+1\}$ be the target function for a binary classification problem.
- Let $D = \{\langle x, f(x)\rangle \mid x \in S$ be the training set (training examples or training samples).

Then we need to compute the approximate target function $\dot{f} : R^n \to \{-1,+1\}$ using D, such that:

$\dot{f}(x) \cong f(x)$ for all $x \in R^n$.

Informally, while experimenting with a new classification algorithm, there are two phases, a training phase and a testing phase. The training phase uses a part of the dataset, called the training samples, to find the approximation function $\dot{f}(x)$. In the testing phase, we apply this function $\dot{f}(x)$ on another part of the dataset, called the testing samples. We then compare the results of the testing phase with the original classes of the testing samples and compute a few measurements to determine the quality of the new classification algorithm.
In clustering, there is no previous information about the data and the classes, which makes the problem

more difficult than classification. The initial number and the identity of the classes is decided first, and then each data sample is assigned to a specific class, based on the nature of the data sample and a specific heuristic procedure such as K-mean [23] .

In this research our interest is classification, because we have previous knowledge about the classes and the data from our previous fMRI experiments [24] [28].

To evaluate a new classification algorithm we need standard measurements to compare the new algorithm with other related algorithms. Common evaluation tools are used for this, such as the accuracy, sensitivity and specificity.

IV. Cellular Automata for Data Mining

A number of CA models useful for classification have been proposed in the literature. To the best of our knowledge, the first of them, Classification Cellular Automata (CCA), was proposed by Kokol et al [22]. It uses a two dimensional CA and a parameter (energy) that changes with time to provide a more accurate classification.

In this model, each feature in the dataset (e.g. age, income, height...) is mapped to a column of the CCA, each column having a predefined threshold. Each cell may be in one of five possible states, four as in fig 2.a, plus the dead state. Depending on their state, they may possess a given "energy" represented by variable "i" in fig 2.a, which may take three values: high (represented by a dark color), low (a light color) or dead (white). The state of a cell informs the learning procedure of the relation between the sample value and the threshold, and whether the training sample was classified correctly.

For example, as shown in fig 2.b, we first select income, then height, then income again (with the thresholds listed for each one) and encode them to the CCA. In more detail, the income value in the first row in the dataset is 14,000 (less than 15,000, the threshold in fig 2.b), so the mapping will be: Xi,j < tj with a low energy (i is false, because it is less by just a little). Whereas the value of income in the second row is 32,000 (greater than 15,000 by a lot), so the mapping will be: Xi,j > tj with a high energy (i is true). In the third row, the income value is 22,000 (greater than 15,000, by not too much), so the mapping will be: Xi,j > tj with a low energy.

The transition rules define the next state by comparing the current state of a cell with the states of its neighbors. The energy of the cell increases or decreases, depending on the actual combination. If a cell energy reaches zero, the cell stops working (is dead). The rules make the bottom training cells disappear after every timestamp. In the classification phase, a single test sample is put at the top of the CCA; then the test sample works its way down the classifiers, until a majority vote can be taken over all the classifiers, as in fig 2.c. The pseudo-code for the CCA algorithm is shown in listing 1.

This approach requires intuition to initialize the values of the energy parameter, and for the threshold selection. The model needs to be redesigned for each dataset. Another disadvantage is that it requires feature selection preprocessing, which is a time consuming operation.

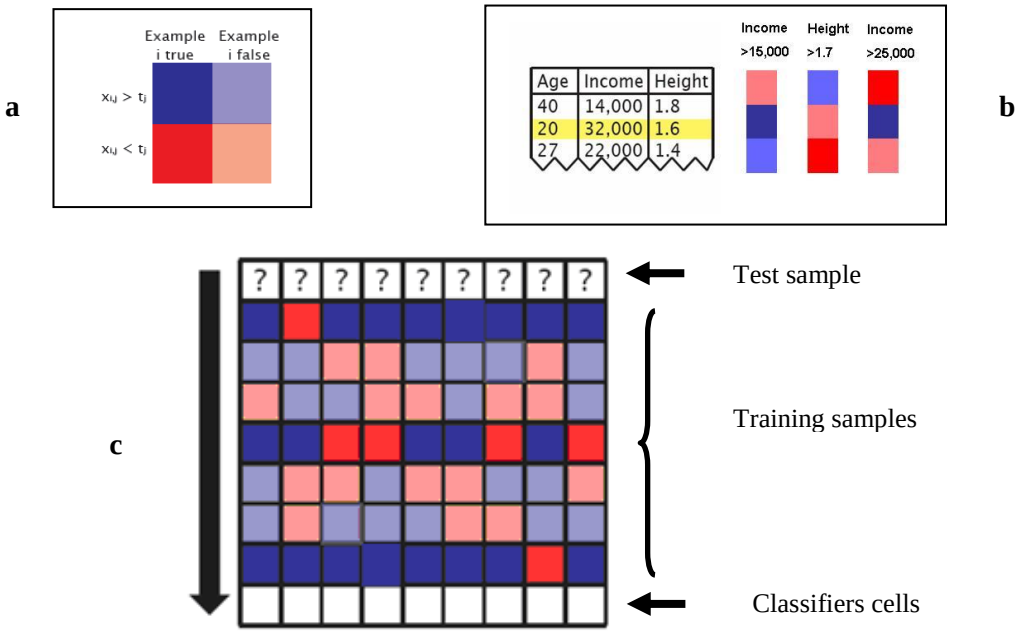


Fig. 2. CCA. a. The four possible states.
b. An example of mapping the dataset to the CCA for three values.
c. The classification process for seven different values.

Listing 1. CCA Algorithm

// learning phase
Input*: 1. Training set with n training samples*
2. Number of iterations T
Output*: CCA with highest classification accuracy on the training set*

1. for t=1 to T
2. choose a learning sample I
3. fill the automaton
4. the cells in the automaton classify the learning sample I
5. change cells energy according to the transition rules
6. cells with energy below zero do not survive
7. end

//Testing phase
Input: A test sample
Number of iterations V
Output: Class of the input sample
1. for t=1 to V
2. the cells in the automaton classify the sample
3. change cells energy according to the transition rules
4. each cell with energy below zero does not survive
5. end
6. Classify the sample according to the weighted voting of the surviving cells.

A different model was proposed by Fawcett in his paper [21]. The CA grid is initiated with training instances (see Fig.3) and the CA is run with a flat space boundary condition. Each cell state represents the class of that point in the instance space, so the cells will organize themselves into regions that have the same class.

each cell's four neighbors and assigns the new state of cell according to the majority class. With this procedure, the class of a given cell may change if the majority class changes.

The advantage of using this CA model is its simplicity, which makes it possible to implement it with hardware and so will run much faster than other data mining methods. The state transition for this model uses a voting rule which assigns a new state to each cell according to the number of neighbors (in a von Neumann neighborhood) in a specific class. A non stable n4_V1 rule is used, which examines The rule is defined thus, for a CA that must classify the date into two different classes (1 and 2):

Non stable n4_V1

0 : class 1 neighbors + class 2 neighbors = 0
1 : class 1 neighbors > class 2 neighbors
2 : class 1 neighbors < class 2 neighbors
Random (Class1, Class2): class 1 neighbors = class 2 neighbors

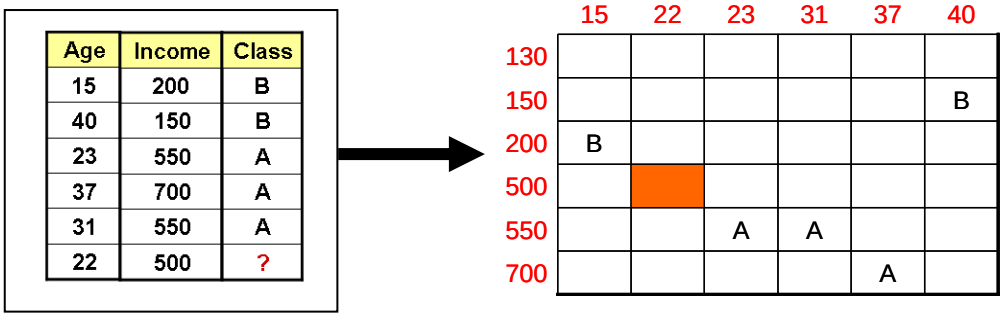


Fig. 3. Mapping the training data-set to CA initial values with Fawcett's CA model.

V. Proposed Algorithm

We propose an enhancement to this model by using the Moore neighborhood, which checks the states of eight cells in all directions. This neighborhood speeds the operation and makes the classification process more accurate. We have also modified the transition rule, so that when the number of class 1 neighbors equals the number of class 2 neighbors, we assign a new class (x) to the cell, as shown in fig 3. This change prevents these cells from changing or affecting the voting process in the next time step. However, at the end of the process, all cells whose state corresponds to the new class are changed randomly to one of the target classes.

modified non stable n4_V1

- 0 : class 1 neighbors +class 2 neighbors=0
- 1 : class 1 neighbors > class 2 neighbors
- 2 : class 1 neighbors < class 2 neighbors
- x : class 1 neighbors = class 2 neighbors

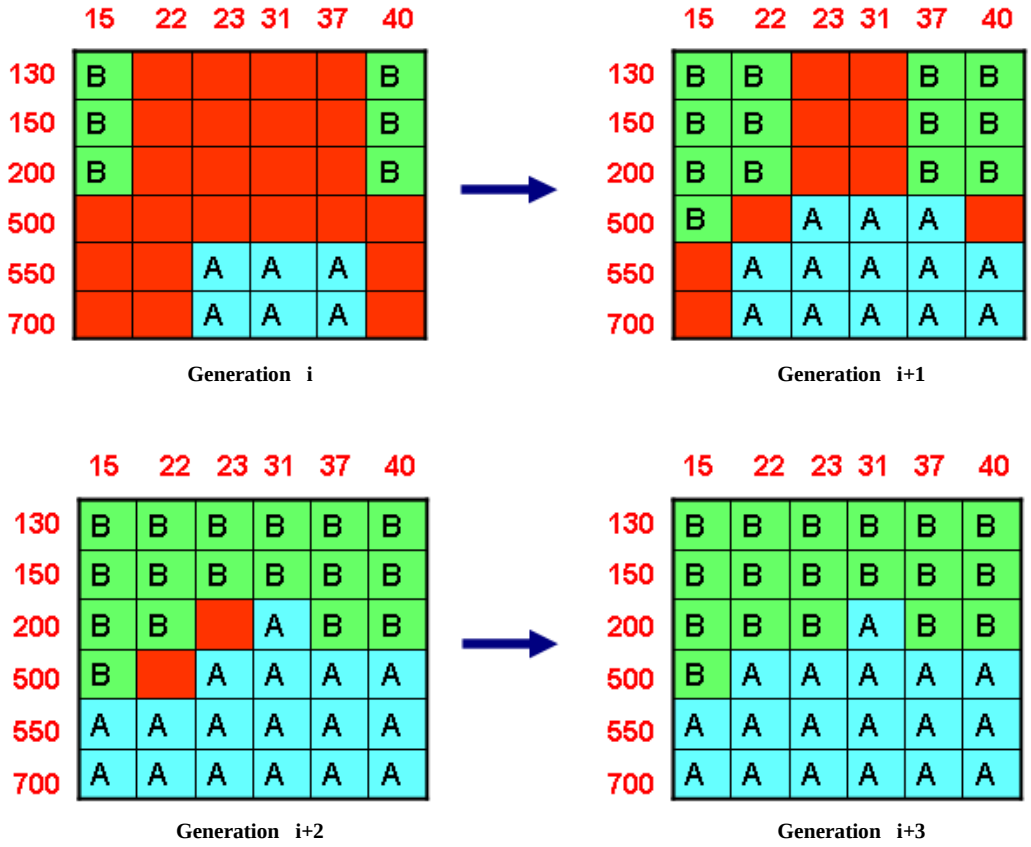


Fig 4: Example of our new model voting process.
a) Cells in blue are assigned to class A.
b) Cells in green are assigned to class B.
c) Cells in brown are currently undecided (they belong to classes 0 or x).

Listing 2: The Algorithm Description

```
Input     : A =Array [r,c] ,each cell represent class of record in the DB
Output  : A with  classified values
Generation=0
While there is a cell with class 0 do
      {  For i=1 to rows_size  do
           For j=1 to columns-size  do
                {
                Check the classes of the 8 neighbors of A[ i,j ]
                If  class 1 neighbors + class 2 neighbors = 0 then
                    A-temp [ i,j ] = 0
                else  If  class 1 neighbors > class 2 neighbors  then
                          A-temp [ i,j ] = class 1
                     else If class 1 neighbors<class 2 neighbors  then
                               A-temp [ i,j ] = class2
                          else          // class1 neighbors = class2 neighbors
                               A-temp [ i,j ] = x  //to avoid being classified again
                }
       A=A-temp        // update the values of A cells
      Generation ++   //Number of generations counter
       }
// Do last loop to update the dump values x
For i=1 to rows_size  do
           For j=1 to column-size  do
                If  A[ i,j ] = x  then
                     A[ i,j ]= random ( 1 , 2 )
Verify the result  //compare A classes with the actual classes
```

VI. Experiments in two dimensions

We have implemented a CA simulator to do experiments with three different models using Borland Delphi Ver.7 and the MS Access database. For simplicity, we experiment with two-dimensional CA with just two classes and three different classification goals, as shown in fig 5.

Each goal is a 32x32 grid (1024 points) of binary data, taken randomly from the data set, so each cell belongs to one of two classes (0 and 1, represented as red and blue in Fig 5). On each experiment, a certain percentage of the data sample (1%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%, 99%) is assigned as initial values to the corresponding (randomly chosen) CA cells. The CA is left to evolve until a final result is reached, where all the cells are classified, and this result is compared to the original data sample.

We compared the four following models:

- The classical K-Nearest Neighbors model [17] with k=9. This value is estimated to correspond to the 8 neighbors in the Moore model.
- Kokol's model
- Fawcett's model, which uses Von Neumann neighborhood.
- Our own model, using Moore neighborhood.

We run the experiments 20 times for each initial value, and then computed the average of the number of generations, the number of error records and the average of the error ratio. So, the total number of

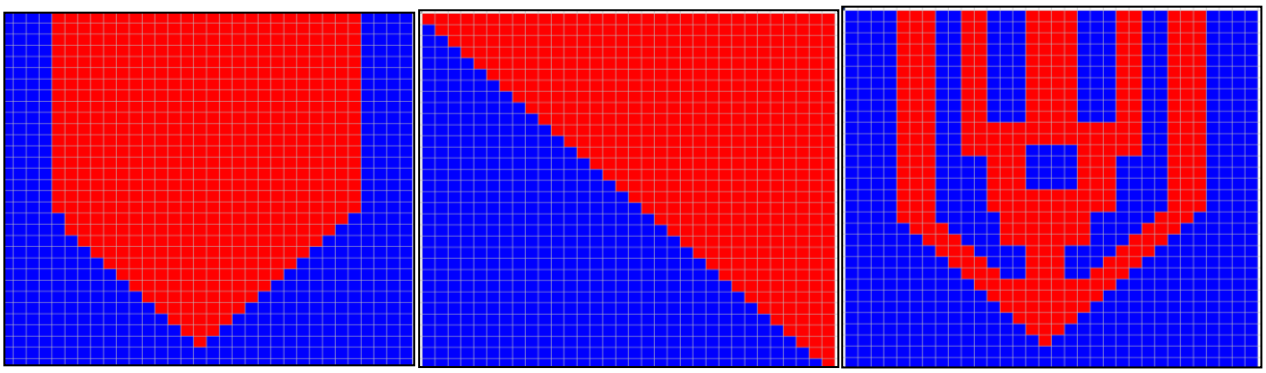


Fig 5. Three Data Samples

experiments for each model and each data sample was 220.

VII. Results

All the experiments we have performed show that our model is superior to the other models, both in terms of accuracy and performance, as can be seen in the tables, graphs and charts shown in Fig. 6 and 7.

- Fig. 6a compares the number of generations needed for convergence in our method and Kokol and Fawcett's methods. It can be seen that ours is better or equal in all cases. In this case, the KNN method is not comparable, as the number of generations is not meaningful.
- Fig. 6b compares the error ratios for the four methods. It can be seen that ours is also better in all cases except for 1% initial values.
- Fig. 6c compares the number of error records in the four methods. Ours is again better in all cases except for 1% initial values.
- Fig. 6d compares the total computation times for the four methods. Except for 1% initial values, ours is clearly competitive with the other methods based on cellular automata. For 30% or more initial values, ours is faster than Kokol's and KNN and as fast as Fawcett's (which is the one we are trying to improve).
- Fig. 7 compares the results of our method with the KNN and Fawcett's methods. It can be seen that ours provides the best approximation to the original figure.

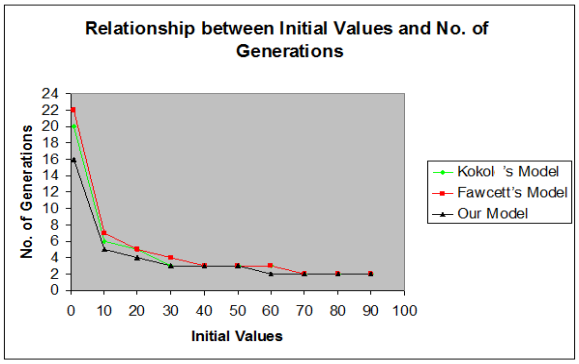


Fig 6 a. Number of generations in the algorithms as a function of % of initial (training) values

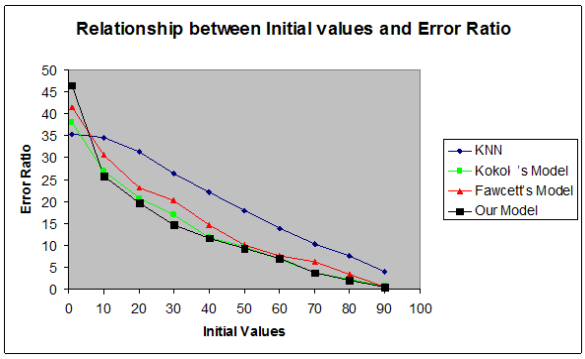


Fig 6 b. Error ratio as a function of % of initial (training) values

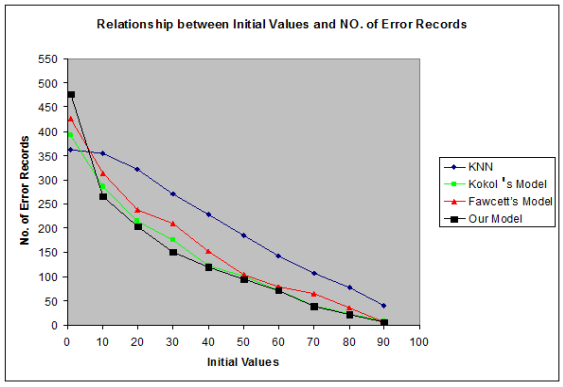


Fig 6c. Nr. of error records as a function of % of initial values

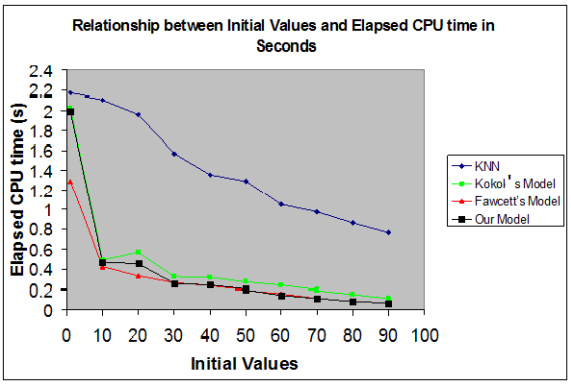


Fig 6d. Computing time as a function of % of initial values

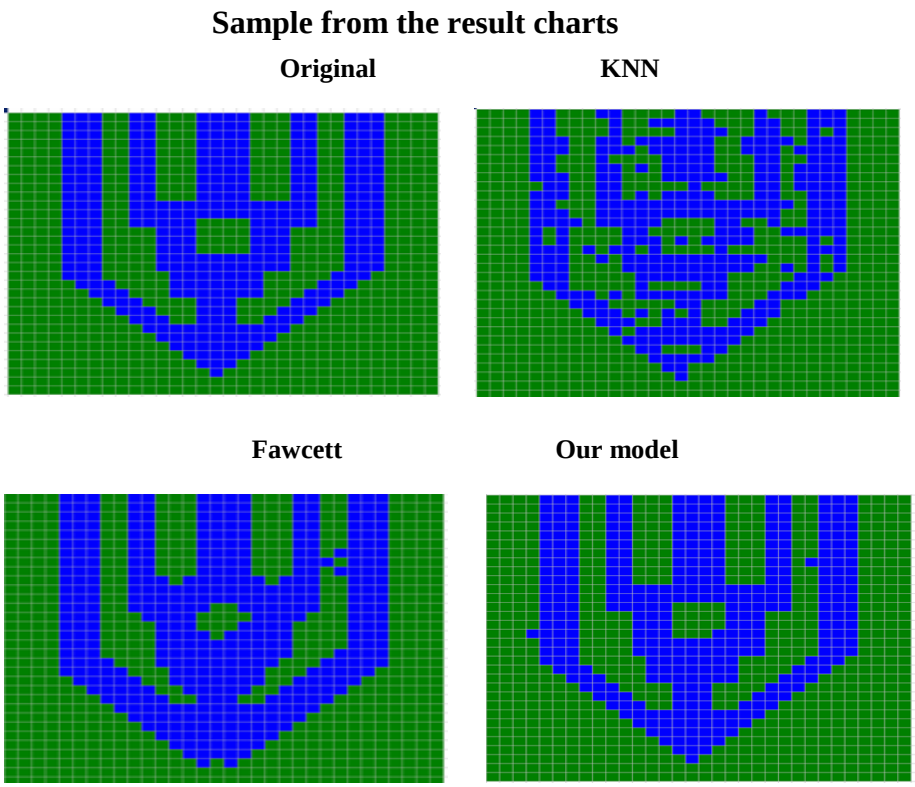


Fig 7. Results Graphs

VIII. Conclusions and future research

In this paper we present an enhancement on the use of cellular automata as a technique for the classification in data mining with higher or the same performance and more accuracy than previous methods. Our enhancements on this technique with respect to Fawcett's method affect the majority rule by using the Moore neighborhood and by introducing a new state for undecided cases.

As a future objective, we intend to analyze the effect of extending the Moore neighborhood to a greater radius. We also suggest that multi-dimensional CAs could be used as a classification technique. The number of dimensions would be correlated with the number of attributes used in the classification process. In general, in d-dimensional space, a von Neumann neighborhood will contain $2^{\mathbf{d}}$ cells. With a Moore neighborhood, it will contain $3^{\mathbf{d}}$ -1 cells. If we have ten attributes (for instance), each with ten records, the work space will consist of 10^{10} cells, and for each cell more than fifty thousand operations would be needed to examine its neighborhood.

References

[1] A cellular automaton model with diffusion for a surface reaction system
Original Research Article Chemical Physics, Volume 165, Issue 1, 1 September 1992, Pages 57-63 ,J. Mai and W. von Niessen

[2] Ermentrout GB, Edelstein-Keshet L, 1993, Cellular automata approaches to biological modeling, Journal of Theoretical Biology, 160, 97-133.

[3]Margolus. N, Toffoli.T, Vichniac. G.1986."Cellular automata supercomputers for fluid –dynamics modeling". Physics Rev. Lett, 56, :1694-1696, 1986.

[4] Boerlijst M. and Hogeweg P.1991. Self-structuring and selection: In Artifial Life II, pp 159:209, ed. By C. G. Langon, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, Addison.Wesley.

[5] Langton G. G. 1984.Self-reproduction in cellular automata. Physica D 10,10:135-144.

[6] Nagel.K and Scheicher,.A ,1994.Microscopic traffic modeling on parallel high performance computers, Parallel Computing, 20,125:146.

[7]A Schadschneider and M Schreckenberg 1993 J. Phys. A: Math. Gen. 26 L679

[8] Slatnia S., Kazar O. (2008) "Generalised Evolutionary Cellular Automata Based-approach for Edge Detection", International

Review on Computers and Software (I.RE.CO.S.), Vol. 3. n. 4, pp. 424 – 428.

[9]Messina, J.P. & Walsh, S.J. 2001. 2.5D Morphogenesis: modeling land use and land cover dynamics in the Ecuadorian Amazon. Plant Ecology, 156: 75-88.

[10]Ward, D. P., Murray, A. T., and Phinn, S. R. (2000) "A stochastically constrained cellular model of urban growth", Computers, Environment and Urban Systems, Vol.24 No.6, 539-558.

[11] Janko Gravner, David Griffeath. (2010) The One-Dimensional Exactly 1 Cellular Automaton: Replication, Periodicity, and Chaos from Finite Seeds. Journal of Statistical Physics Online publication date: 10-Dec-2010.

[12] Lindgren .K and Nordahl. M,1990. Universal computation in simple one dimensional cellular automata, Complex Systems, 4 , 299-318.

[13] Morita, K., and Ogiro, T.: Simple universal reversible cellular automata in which reversible logic elements can be embedded, IEICE Trans. on Information and Systems, E87-D, 650-656, 2004.

[14]Offoli. T,1977. Computation construction universality of reversible cellular automata, J. Comput. Syst. Sci., 15 :213-231.

[15] Nordahl. M.G,1989.Formal languages and finite cellular automata.Complex systems, 3:63-78,1989.

[16] Culik II.K, Hurd .L.P, Yu. S, 1990."Computation theoretic aspects of cellular automata". Physica D,45:396-403,1990.

[17] Kantardzic, M. (2003) Data mining: Concepts, models, methods and algorithms, USA: John Wiley and Sons

[18] Daniel T. Larose, Data mining methods and models , Wiley IEEE Press,2006.

[19] David Hand, Heikki Mannila, and Padhraic Smyth. Principles of Data Mining. (2001) MIT Press, Cambridge, Massachusetts. ISN 0-262-08290-X

[20] Abraham Silberschatz, Henry F. Korth, S. Sudarshan. Database Systems Concepts Publisher: McGraw-Hill Higher Education 2006 , 1170 Pages ,ISBN: 0072958863

[21] Tom Fawcett .; Data mining with cellular automata , SIGKDD Explorations.; Volume 10.; Issue 1.pp:32-39, July 2008.

[22] P. Kokol, P. Povalej, M. Lenic and G. Stiglic, "Building classifier cellular automata", Cellular Automata. 6th International Conference on Cellular Automata for Research and Industry, ACRI 2004, Holanda, Springer-Verlag, Lecture Notes in Computer Science Vol. 3305, pp. 823–830, 2004.

[23] Witten, I. and Frank, E. (2011), Data Mining Practical Machine Learning Tools and Techniques, 3rd edition, Elsevier Inc. 2011.

[24] Abu Dalhoum, A.L. and Al-Dhamari, I. (2010), fMRI Brain Data Classification using Cellular Automata, New Aspects of Applied Informatics, Biomedical Electronics, and Informatics and Communications, 2010.

[25] Alfonseca.M, Dalhoum. A.Abu, Ortega. A, 2001
" Evolving the game of life with a genetic algorithm", Proceedings of the 3rd Middle East Symposium on Simulation and Modelling (MESM'2001), SCS Publications, p.165-169, ISBN: 1-56555-230-X.

[26] Conway,J.H., Berlekamp,E.R., Guy,R.K.,Winning ways for the mathematical plays. London: Academic Press, Vol 2,Ch.25.

[27] Jecheva V. G., Nikolova E. P. (2008), "An Adaptive Approach to Anomaly Intrusion Detection Based on Data Mining and String Metrics. International Review on Computers and Software (I.RE.CO.S.), Vol. 3. n. 5, pp. 515 – 522.

[28] Sleit, A., Abu Dalhoum, A.L., Al-Dhamari, I. and Awwad, A. (2009), Efficient enhancement on cellular automata for data mining, Proceedings of the 13th WSEAS international conference on Systems, ICS'09.

Authors' information

[1] Department of Computer Information Systems, King Abdulla II School for Information Technology, University Of Jordan, P.O. Box 13898, Amman 11942, Jordan
[2] Department of Mathematics and Computer Science, Faculty of Science, TafilaTechnical University, Tafila, JORDAN
[3] Departamento de Ingeniería Informática, Escuela Politécnica Superior, Universidad Autónoma, Madrid, SPAIN

Omar Y. Adwan received his BSc in computer science from Eastern Michigan University, USA MSc and PhD in Computer science from George Washington University, USA, majoring in Software Engineering. Currently he is the chairman of Computer Information Systems at the University of Jordan. His research interests include Software Engineering, Information Security, Quality Assurance, Software Architecture, and Information Systems Audit.

Ammar M. Huneiti received his BSc, MSc and PhD degrees from Cardiff University, UK. His BSc is in Computer Science (1991), his MSc is in Information Systems Technologies (1992) and his PhD is in Systems Engineering (2004). Between 1992 and 2000 he worked for several private and public sector organizations supervising the design and implementation of IT related projects. Since 2005 and until present, he is an assistant professor at the Department of Computer Information Systems, King Abdullah II School of Information Technology, the University of Jordan. His research interests include Intelligent Web Information Systems, Web Data Mining, Adaptive Hypermedia, and User Modelling.

Abdel Latif Abu-Dalhoum received his BSc. degree from AL Mousel University, Iraq, M.S. and PHD degrees from University Autonoma of Madrid, Spain in 2004. Currently he is Associate Professor in Jordan University, Jordan. His current research interests include artificial life, genetic algorithms, cellular automata, DNA computing